

## Research Article

# Transmission Rate Sampling and Selection for Reliable Wireless Multicast

Thomas Geithner  and Fikret Sivrikaya 

Department of Electrical Engineering and Computer Science, Technische Universität Berlin, Ernst-Reuter-Platz 7, Sekr. TEL 14, 10587 Berlin, Germany

Correspondence should be addressed to Thomas Geithner; [thomas.geithner@tu-berlin.de](mailto:thomas.geithner@tu-berlin.de)

Received 9 March 2020; Revised 21 May 2020; Accepted 22 July 2020; Published 30 September 2020

Academic Editor: Thomas Newe

Copyright © 2020 Thomas Geithner and Fikret Sivrikaya. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The multicast communication concept offers a scalable and efficient method for many classes of applications; however, its potential remains largely unexploited when it comes to link-layer multicasting in wireless local area networks. The fundamental lacking feature for this is a transmission rate control mechanism that offers higher transmission performance and lower channel utilization, while ensuring the reliability of wireless multicast transmissions. This is much harder to achieve in a scalable manner for multicast when compared with unicast transmissions, which employs explicit acknowledgment mechanisms for rate control. This article introduces *EWriM*, a reliable multicast transmission rate control protocol for IEEE 802.11 networks. It adapts the transmission rate sampling concept to multicast through an aggregated receiver feedback scheme and combines it with a sliding window forward error correction (FEC) mechanism for ensuring reliability at the link layer. An inherent novelty of *EWriM* is the close interaction of its FEC and transmission rate selection components to address the performance-reliability tradeoff in multicast communications. The performance of *EWriM* was tested in three scenarios with intrinsically different traffic patterns; namely, music streaming scenario, large data frame delivery scenario, and an IoT scenario with frequent distribution of small data packets. Evaluation results demonstrate that the proposed approach adapts well to all of these realistic multicast traffic scenarios and provides significant improvements over the legacy multicast- and unicast-based transmissions.

## 1. Introduction

Multicast in IP networks is a well-established concept, but it still plays only a marginal role within local networks, being mainly used for support functions (like network management and service discovery). With today's development of Internet applications, multicast gains more relevance. On the one hand, IP networks become more and more important for all kinds of multimedia content (Internet services as well as local applications). On the other hand, new types of applications with group-based communication models are growing, for example, in the domain of Internet of Things (IoT), which triggers new impulses on multicast. Examples for this development are multicast-enabled versions of CoAP [1] and MQTT [2].

In many cases, wireless network technologies have already become the default way of connecting devices to the Internet infrastructure, and this trend can be expected to go on further. Wireless networks, which exhibit a natural broadcast characteristic due to the shared medium, fit very well to the multicast communication concepts, at least in theory. However, in practice, exploiting the benefits of this concept usually comes with unique challenges for different radio technologies.

In IEEE 802.11 networks, multicast has only been supported at an elementary level since the beginning, providing neither reliability nor high performance. This is sufficient for the common network management and support functions, but prevents more advanced use cases for multicast. Later standard improvements have addressed the most

critical issues and provided some workarounds to provide better performance or reliability for multicast traffic. But up until now, there is no satisfying solution bringing the potential of the multicast concept to IEEE 802.11 networks, especially in terms of resource efficiency and performance.

The essential problem in terms of performance is the transmission rate selection. In the case of multicast, the problem itself is more complex than on unicast, since one transmission rate must be selected that fits all receivers in a specific group. Moreover, the acknowledgment mechanism, which is the basis of common unicast rate controls, is not available on multicast because it would cause collisions on simultaneous responses from all receivers, in addition to the large communication overhead.

In this article, we present an approach aiming at enabling the full potential of the multicast concept for 802.11 networks, which could significantly improve the efficiency and performance. Unlike most other multicast transmission rate control concepts, we adapt the sampling mechanism, which is commonly employed in unicast, for multicast communication. The inevitable packet losses caused by the rate sampling are handled by a flexible forward error correction (FEC) mechanism, which also addresses the general reliability issue of multicast transmissions. An important novelty of the presented approach is the close interaction between the rate sampling mechanism and the FEC layer, which allows applying the former concept on MAC layer multicast transmissions. The protocol proposed in this article, *Enhanced Rate Control for Wireless Reliable Multicast (EWRiM)*, is aimed at general multicast network applications and is not limited to specific use cases like multimedia content distribution.

In the rest of the article, after giving an overview of the related work in Section 2, we describe our approach in detail (Section 3), followed by selected operation examples (Section 4) and comparative evaluations in different scenarios (Section 5), illustrating the improvements over the legacy multicast mechanism as well as over unicast. Finally, Section 6 concludes the article with a summary and future research directions.

## 2. Related Work

The IEEE 802.11 standard series had been optimized for unicast data traffic since its early days, while broadcast and multicast data only played a supplementary role. Because group communication inhibits immediate acknowledgments as they are used for unicast data frames, the default MAC behavior waives any feedback mechanisms for broadcast and multicast data frames. This not only affects reliability, since no retransmissions are initiated after a frame loss, but also prevents common rate control algorithms being applied to multicast data. Both issues have been addressed in the past and some improvements became part of newer IEEE 802.11 standard versions, but the handling of multicast data is still far from its potential.

**2.1. MAC Layer Reliability for Multicast.** Achieving reliability without reducing scalability is a general challenge in multicast communications. Two types of approaches are com-

monly used for addressing this problem: *automatic repeat request (ARQ)* mechanisms and *forward error correction (FEC)* methods.

Since the ARQ approach relies on a feedback mechanism (similar to the unicast behavior), the challenge is enabling it for the multicast case without suffering from scalability issues as the receiver group size grows. One concept is the reduction of feedback messages. Using negative (NAK) instead of positive (ACK) acknowledgments is a common approach for multicast at different protocol layers. But without additional mechanisms, it does not solve the collision problem in the 802.11 MAC layer when it is used on a dedicated time slot similar to unicast ACKs. Another way of reducing the amount of feedback messages is the *leader-based approach*, where one representative node is selected for providing feedback information, either in the traditional way with positive ACKs or in combination with NAKs. Since the receiver nodes might face different channel conditions with regard to the sender (e.g., distances and thus signal strengths), the leader should represent the worst conditions among the group, which makes the leader selection an essential but also difficult task. The *Leader-Based Protocol (LBP)* [3] is one example for this approach using a combination of ACKs and NAKs.

The use of collision-resistant feedback mechanisms is also proposed in the literature, e.g., through the utilization of *tones* (narrow-bandwidth sinusoidal wave signals) instead of digital frames, which allow simultaneous signaling. Out-of-band tones are used by Gupta et al. [4] for two purposes, a *busy tone* indicating that a node is currently not able to receive any data and a *NAK tone* signaling a packet loss. A substantial drawback of this solution is the need of additional hardware components for the out-of-band signaling. Another approach is simultaneous feedback on separated bands using the 802.11 PHY instead of a dedicated hardware. As an example, *HIMAC* [5] transmits unary signals (similar to tones) on dedicated OFDM subcarriers that are assigned to the receiver nodes. Using the 802.11a/g PHY, this allows simultaneous signaling from up to 17 receivers. HIMAC uses two unary signals, a short NAK signal (*UNF*) and a report of supported transmission rates using the signal length for encoding the information (*UCF*). Similarly, *OMACK* [6] utilizes dedicated subcarriers for receiver feedback, but with digital encoded data instead of unary signals for individual ACK messages. *FlexVi* [7] uses OFDM subcarriers as a bit-map carrying NAK signals for a block of frames. Sequential transmission of individual feedback messages also helps avoid collisions, although it results in performance degradation. *BMMM* [8] does this by individual polling with one pair of *Request for ACK (RAK)* and ACK frames per receiver node. In *RMAC* [9], a dedicated time slot is assigned to each receiver before transmitting a data frame, which is then acknowledged by each receiver with short *Acknowledgment Busy Tones*, requiring less time than a frame.

Instead of acknowledging individual frames immediately, aggregated feedback information is aimed at reducing the overhead. The IEEE 802.11e [10] amendment introduced the *Block ACK* mechanism for unicast data focusing on improving the throughput. This allows sending feedback

information for a whole block of data with a single frame upon request by the receiver. In IEEE 802.11a [11], it has been extended to multicast data with the *Groupcast with Retries Block ACK (GCR-BA)* mechanism where the sender polls the individual from all receivers sequentially.

As an alternative to postreaction to lost frames, forward error correction adds redundancy into the data flow aiming at avoiding the need for individual feedback information. Traditional block codes (e.g., *Reed Solomon*) use a fixed block size of  $k$  source symbols (e.g., frames) and extend this to  $n$  code symbols by adding  $n - k$  redundancy symbols. A receiver can reconstruct all source symbols if he received at least  $k$  frames of a block. The code rate  $R$  is defined as  $R = k/n$ . On a given code rate, longer blocks provide higher flexibility for recovering lost information. For example, a code with  $k = 2$  and  $n = 3$  ( $R = 2/3$ ) cannot recover two consecutively lost symbols whereas a code with  $k = 4$  and  $n = 6$  can do this at the same code rate. But this comes at the price of higher decoding delay. Thus, applying this traditional block codes requires a choice between powerful recovery (longer blocks) and low delay (short blocks).

A newer approach is coming from the field of *Network Coding (NC)* (where usually the term *generation* is used instead of *block*). While *Network Coding* focuses mainly on other aspects, it also provides a powerful and flexible FEC mechanism. In *Linear Network Coding*, coded symbols form linear combinations of  $k$  source symbols. Decoding the source symbols requires at least  $k$  code symbols and, in addition, their individual coefficient vectors. Since *Network Coding* also addresses use cases like multipath routing and multipath fading in wireless multihop networks, the usage of random coefficient vectors is the commonly used variant (*Random Linear Network Coding (RLNC)*), since it simplifies the coefficient vector management in distributed environments. But this requires the transmission of additional information for decoding the symbols (e.g., the coefficient vector or a seed value as proposed in [12]) as part of a code symbol.

Based on this mechanism, a *sliding window FEC* [13] provides a more flexible approach of dealing with the tradeoff between powerful recovery and delay. Instead of creating the code symbols using only one block, they are computed over a larger encoding window covering several blocks. This kind of interleaving allows the recovery of small losses with a low delay (e.g., when  $k$  symbols of the current block have been received) while larger consecutive losses can be recovered if sufficient symbols of the following blocks reached the receiver.

Besides the already mentioned *GCR-BA*, the *aa*-amendment of the 802.11 standard also introduced the *GCR-UR (Unsolicited Retries)* mode, on which multicast frames are transmitted multiple times proactively for achieving higher reliability. This can be seen as a very simple form of forward error correction, but stays far below the power of coded packet transfers.

**2.2. Multicast Transmission Rate Selection in 802.11 Networks.** An essential part of the 802.11 MAC layer is the transmission rate control, which is mandatory for efficient use of the radio channel resources. For achieving this goal,

it needs to adapt dynamically to the current channel conditions and select an appropriate transmission rate that provides an appropriate tradeoff between performance, resource efficiency, and reliability. Since channel conditions are specific to each receiver, the rate control relies on individual feedback information.

Two different approaches are generally adopted in the literature. The concept of using the measured *Signal-to-Noise Ratio (SNR)* or *Signal-to-Interference-plus-Noise Ratio (SINR)* for selecting the transmission rate is based on the strong theoretical correlation between SNR, *modulation and coding scheme (MCS)*, and *bit error rate (BER)*. A rate control protocol following this approach reports either the receiver SNR back to the sender or the transmission rate selected by the receiver based on the SNR. Even though this approach is often used in theoretical studies, it suffers from some practical issues. A major drawback of this approach is that it does not reflect other aspects affecting the packet transmission success, for example, collisions and transient interferences. Additionally, measuring the SNR is a difficult task since it can vary quickly in time (even during a single frame) and with the frequency band among the individual subcarriers. Hence, the SNR on its own is mainly seen to be insufficient for practical rate selection in 802.11 networks [14].

Most of the practically relevant rate control implementations follow another concept by using the transfer success, for example, in the form of the *packet delivery ratio (PDR)*, as the input for rate selection. On unicast data, the common method for this is using the acknowledgment frame following immediately after the data frame. Because this feature is not available on broadcast and multicast data frames, they are usually transmitted on the lowest data rate assuming that this provides the highest coverage and reliability.

Since the general problem of a multicast-enabled feedback mechanism is the same as those already discussed on the reliability problem in Section 2.1, many multicast rate control approaches follow the similar concepts. While *HIMAC* [5] and *FlexVi* [7] provide collision-tolerant feedback mechanisms, they are using simple SNR-based rate selections. An extension of *OMACK* is presented in [15], where the sender announces the planned data rate in RTS frames. The receivers measure the SNR on the RTS reception and use dedicated OFDM subcarriers for small feedback messages, allowing a correction of the sender's choice. The rate selection itself is done on predefined SNR thresholds. *ARSM* [16] combines the leader-based feedback approach with an SNR-based rate selection.

Over time, the 802.11 standard series were extended by some mechanisms related to multicast, which partially address the transmission rate selection problem. The *v*-amendment, which became part of the 2012 version [17], introduces the *Directed Multicast Service (DMS)*, which translates multicast flows into separate unicast streams on the MAC layer. This provides a suitable workaround for small receiver groups at the expense of lost scalability. It also defines methods for multicast group management (which is required for DMS) and with *Flexible Multicast Service (FMS)*, a simple mechanism for receivers to report usable

transmission rates among which the sender can choose. As mentioned earlier, the *aa*-amendment (which was integrated into the 2016 version [18]) introduced the *Groupcast with Retries* (GCR-BA) mode. Since this provides feedback information about the packet transfer success, it can be utilized for rate selections as well.

Based upon these mechanisms, some multicast rate control approaches have been proposed. Different rate adaption algorithms using the GCR-BA feedback information are compared in [19] based on simulation results. Transmission rate selection and mobility management are addressed as a joint problem in [20, 21]. Whereas *Software-Defined Networks* (SDN) are used for the mobility management part, the rate selection uses a periodic switching between DMS and the legacy multicast transmissions on the wireless part. The DMS period allows an individual rate selection for each receiver using any unicast rate control mechanism. In the following legacy period, these individually selected rates are used for choosing the highest common rate for the whole group.

Real-time video is one of the most prominent use cases for multicast transmissions in general and at the same time quite a challenging one for 802.11 networks since its data rate can easily exceed the available capacity of traditional multicast handling. A special purpose rate control mechanism for video streams has been presented with *MuDRA* [22]. It uses a subset of receivers for collecting feedback information and tries to achieve a PDR of  $\geq 85\%$  for at least 95% of the receivers. The *target rate* is estimated based on the reported PDRs. It is assumed that the transmission rate can be increased as long as the PDR is above a threshold of 97% on most nodes.

Similarly, *InFRA* [23] tolerates some packet loss (PDR of 99% on 95% of the receivers). For achieving this, *InFRA* includes a FEC mechanism that adapts the code rate dynamically. The transmission rate selection is mainly done based on the RSSI (Received Signal Strength Indicator), which is assumed to be coupled to the SNR (but might be specific to the different hardware implementations). Since a packet loss can be caused by a low SNR or interference, *InFRA* tries to distinguish between channel errors and different types of interferences (weak and strong) by using the RSSI and the reported CRC errors. Depending on the estimated cause of a packet loss, *InFRA* reacts differently by either decreasing the transmission rate or increasing the code rate.

One basic mechanism of common unicast rate controls, for example, minstrel [24], is *sampling*, which provides information about the PDR on other transmission rates than the currently selected one. This works efficiently in unicast, since the 802.11 MAC contains a retransmission mechanism (usually with up to 6 retransmits) triggered by missing ACK frames. Based on that, a unicast rate control can safely try other transmission rates with high risk of losses in order to gain knowledge about the current channel conditions. The concept of sampling is not widely used for multicast rate control approaches because it would require an efficient packet loss handling which is, as explained before, a difficult issue.

In our previous work, we presented an experimental multicast rate control approach [25] that used sampling in conjunction with aggregated receiver feedback but did not

provide full reliability at the link layer. Instead, a controlled level of packet losses was delegated to the transport layer, where the NORM protocol [26] was used to ensure reliable multicast. The work that we present in this article extends our previous work by ensuring reliability at the link layer by employing FEC and provides further improvements on the rate sampling and feedback collection mechanisms.

### 3. Enhanced Rate Control for Wireless Reliable Multicast (EWRiM)

Our approach is aimed at improving the performance and reliability of multicast in 802.11 networks while keeping the scalability, which is the key asset of the multicast concept. It consists of three interacting parts, each of which will be described in the following subsections: (i) active group membership management, (ii) sliding window forward error correction, and (iii) transmission rate sampling and selection. These components and their interaction are illustrated in Figure 1.

**3.1. Active Group Membership Management.** An important precondition for efficient multicast is *demand awareness*, which enables a sender to decide if a multicast flow needs to be transmitted to a certain network segment. On the IP layer, this is done via *IGMP* (*Internet Group Management Protocol*) between the group members and the responsible router. In the general case, router functionality and 802.11 stack are working independently of each other and can be located at different devices. *IGMP Snooping* is a method on L2 devices (e.g., Wi-Fi access points, 802.3 switches) for gaining group membership knowledge by passively listening to IGMP messages exchanged between nodes and routers.

Even though this improves the situation, it is not sufficient for our needs. On the one hand, there is a certain chance that an 802.11 node misses some IGMP frames (especially since these are also multicast messages, which lack L2 reliability). More important is that *IGMP Snooping* does not provide a solution for handover between different access points in the same subnet (where no new L3 membership report is required). So we adopt a simple active group management mechanism, where each node periodically broadcasts its group memberships, which can be tracked by its neighbors. In case of an unintended connection loss, a repeated missing renewal leads to the expiration of the membership information on its neighbor nodes. For efficiency reasons, all group memberships are sent within a single frame rather than individual transmissions on *IGMP*.

In addition to the group membership information, these messages are used for exchanging the supported transmission rates among nodes in case this is not already done in other parts of the 802.11 stack (e.g., when the ad hoc mode is used). This allows selecting the set of commonly supported transmission rates  $T$  for a multicast group  $G$ . For avoiding consecutive collisions of messages from different nodes, the period for group membership messages contains a random offset. Any other solution providing the same information (group memberships and supported transmission rates) could also be used as an alternative. In particular, the *Flexible*



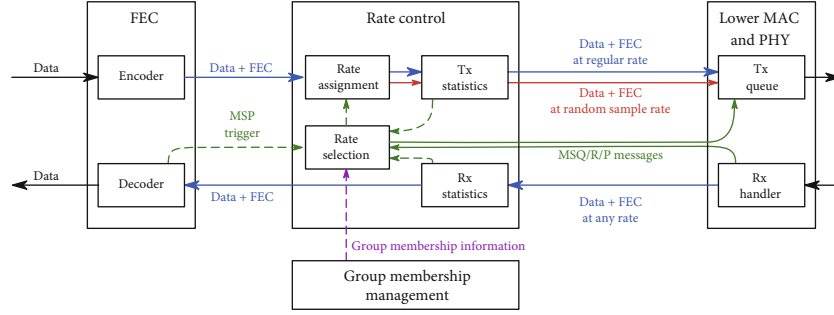


FIGURE 1: High-level overview of the components and their interactions in EWRiM.

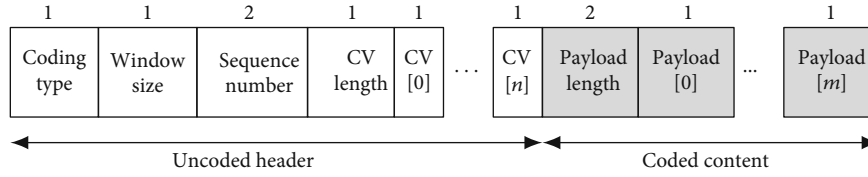


FIGURE 2: Intermediate FEC protocol header.

*Multicast Service (FMS)*, introduced in the *v*-amendment and integrated into the 2012 version of the standard [17], would be a viable option. For simplicity and because FMS was not implemented in the simulation framework, we use our own implementation for this purpose without affecting the general rate control concept described in the following.

**3.2. Sliding Window Forward Error Correction.** A major drawback of multicast transmissions in 802.11 radio networks is the lack of an integrated ARQ mechanism, which leads to reduced reliability compared to unicast. This aspect becomes even more important in our sampling multicast rate control (explained in Section 3.3) due to additional packet losses. Because the necessary acknowledgment cannot easily be adapted to multicast, *forward error correction (FEC)* seems to be a more promising approach, which offers a mechanism for recovering lost frames without the need of feedback messages.

Our approach integrates an intermediate FEC layer, which handles all multicast data frames, into the 802.11 stack, providing the sliding window coding mechanism. In our implementation, we are using the *Kodo Slide* library [27], a sliding window erasure correcting code (ECC) implementation based on the *Kodo* library [28]. The coding is performed individually per flow (identified by group and source MAC addresses) since the receiver groups might differ among the flows. This protocol layer resides between the LLC (Logical Link Control) and the 802.11 MAC and includes a short header containing information required for decoding (Figure 2). Currently, only two coding types are used, uncoded frames and RLNC coded frames using a Galois Field ( $2^8$ ). The window size is required to be known at the receiver for maintaining a decoding buffer of sufficient size. Each source frame of a flow is identified by a sequence number. Coded frames carry the sequence number of the latest source frame. Since redundancy frames do not include a new source frame, they carry a sequence number that had

already been used by another frame before (as explained in [13]). In case of a coded frame, the header additionally contains the required coefficient vector. Its length should normally be equal to the window size, but it can be shorter (e.g., on start of a flow, when the encoding buffer is not completely filled). This protocol header is considered as a proof of concept for our approach; some improvements or alternatives are possible (e.g., implicit window size determination using the coding vector length or transmitting only a seed value instead of the complete coding vector [12]).

RLNC allows the mixing of coded and uncoded frames. This feature can be used for *systematic coding*, where source frames are sent uncoded and complemented by coded redundancy frames (which is comparable to traditional FEC methods). For other use cases, it can be beneficial to send all frames coded, especially in multihop scenarios in combination with recoding for taking advantage of multipath routing. A constraint of coded frames is that their size has to cover the maximum length of their included source frames. In our implementation, the coded frame size is adapted dynamically to the maximum length within the encoding window and shorter frames are padded. This is not a drawback on flows of equal-sized frames, but causes an additional overhead on non-equal-sized flows. In the second case, systematic coding can be more efficient, since the uncoded frames are not required to be padded. For reconstructing the source frames correctly, the frame length is included into the coded part of the frame (as shown in Figure 2).

In our environment, we are using a static window size and a fixed scheme of redundancy frames. But as a further improvement, both could be managed dynamically. In particular, the amount of redundancy frames could be adapted to the current conditions based on the already available or additional feedback information.

On decoding, the reconstructed source frames are delivered in order of their sequence numbers. Incompletely decoded frames remain in the decoder buffer until sufficient

information has been received. The *decoder queue length*, in our definition, is the number of frames in the decoder buffer, which could not yet be decoded. This includes also complete (esp. uncoded) frames in the buffer, if an incomplete one before still blocks their decoding. When all available frames have been successfully decoded, the queue length is 0. If a frame could not be decoded before exceeding the buffer size, it is dropped and the decoding process can continue with the following frames. The current decoder queue length can be used as an indicator for the actual channel situation as it is caused by lost frames that have not been reconstructed. So, we are using this queue length as a trigger for our rate control (Section 3.3) when it reaches a configured threshold.

This queue-length-based trigger is a complementary mechanism that provides faster reaction than the primary polling in case of high packet loss. As a future improvement of this for low-traffic scenarios, it could be supported by a timeout smaller than the regular polling interval, which triggers the rate control in case of uncoded packets remaining in the queue for a certain duration. That could improve the behavior on low-frequency data, but since the choice of an appropriate timeout value would depend on the traffic pattern, this mechanism should adapt automatically to it.

This FEC layer provides an effective and flexible way for compensating the packet loss at the transmission, but it comes at the price of extra overhead on the radio channel for the additional redundant packets as well as the computation effort at the sender and receiver for encoding and decoding the packets.

The network overhead can easily be calculated by the increased packed size for the additional FEC protocol header and the code rate for the additional packets. It should be noted that the overhead estimation in the unicast case is more complex due to possible multiple retransmissions and depends on the channel conditions and the used unicast rate control algorithm. Since the overhead in terms of channel utilization depends on the selected transmission rates, which then results in a specific channel allocation time, a general statement cannot be given without taking the rate control into account. In the later sections, we therefore measure the channel utilization covering all involved parts.

In terms of the computing effort, it has to be differentiated between encoder and decoder. The encoding effort grows linearly with the FEC window size whereas the decoding effort scales generally with its third power. A more detailed analysis of this can be found in the description of *Caterpillar RLNC* [13]. Since the computing effort for coding is mainly caused by matrix operations, implementations can benefit from capabilities of modern CPUs including their SIMD processing extensions. Practical implementation aspects and performance measurements for conventional (generation based) RLNC are presented in [29]. Furthermore, the computing effort can be reduced by using systematic coding. This requires the encoding effort only for the redundancy packets and the decoding effort only for recovering lost frames.

**3.3. Transmission Rate Sampling and Selection.** Our multicast rate control approach combines the concept of data rate sam-

pling (which is the common approach for unicast rate controls like *minstrel* [24]) with an aggregated receiver feedback mechanism as we introduced in our previous work [25]. In contrast to our previous approach, the receiver feedback information is polled individually in a round robin manner for all members in the multicast group to avoid collisions of response frames. This mechanism can be additionally improved by using the reported receiver SNR for a more selective sampling.

**3.3.1. Sampling.** As an essential part of the multicast rate selection, the sampling process provides the required data for calculating the *packet delivery ratio* (PDR) of all available transmission rates. For this, a small share (e.g., 10%) of the data frames is sent with other rates than the currently selected one. We take into account that some of those sample frames will probably not be received by all group members and, thus, be lost. In our concept, this should be compensated by the FEC layer described in Section 3.2. Therefore, the share of sampling frames is adjusted to be smaller than the redundancy provided by the FEC.

For a given share of sampling frames, the transmission rate is selected uniformly at random among the rates commonly supported by the group. Optionally, this is restricted by the minimum reported SNR among the receivers. Since the SNR measurement has some known uncertainties, a configurable tolerance range (e.g., 3 dB) can be added. This SNR-supported sampling provides more valid data in case of low SNR conditions, since it avoids sampling on unpromising rates.

**3.3.2. Rate Selection.** The core part of our approach is the rate selection based on the aggregated receiver feedback information. This requires the sender and the receivers counting the frames per transmission rate in the vector  $\vec{s}$ , respectively,  $\vec{r}^g$ , where  $g \in G$  represents the multicast group member. (Nomenclature contains a summary of all symbols used throughout the article.) An element  $t \in T$  of these vectors, denoted as  $\vec{s}(t)$ , respectively,  $\vec{r}^g(t)$ , contains the number of sent, respectively received, frames at transmission rate  $t$ . For referring to the state of the counter vectors at a specific moment, we use a subscript notation, for example,  $\vec{s}_i$ , where  $i$  represents the time or sequence index. Each time a multicast frame passes the transmission rate control, the corresponding sender counter vector  $\vec{s}$  is updated. The same applies to  $\vec{r}^g$  on successfully received frames at the multicast group members.

In addition to the counter vectors, the reported SNR of the received frames is used, if available, for calculating the average received SNR using *EWMA* (*exponentially weighted moving average*) smoothing for the flow. Similar to the FEC mechanism, the rate control acts per flow. This means, in particular, that all data including the counter vectors are managed individually for each flow.

For collecting the receiver counters, the sender transmits a *multicast statistics query* (MSQ) frame to the receivers containing a sequence number  $i$ , as shown in Figure 3(a). At the moment the MSQ is generated, the sender keeps the current

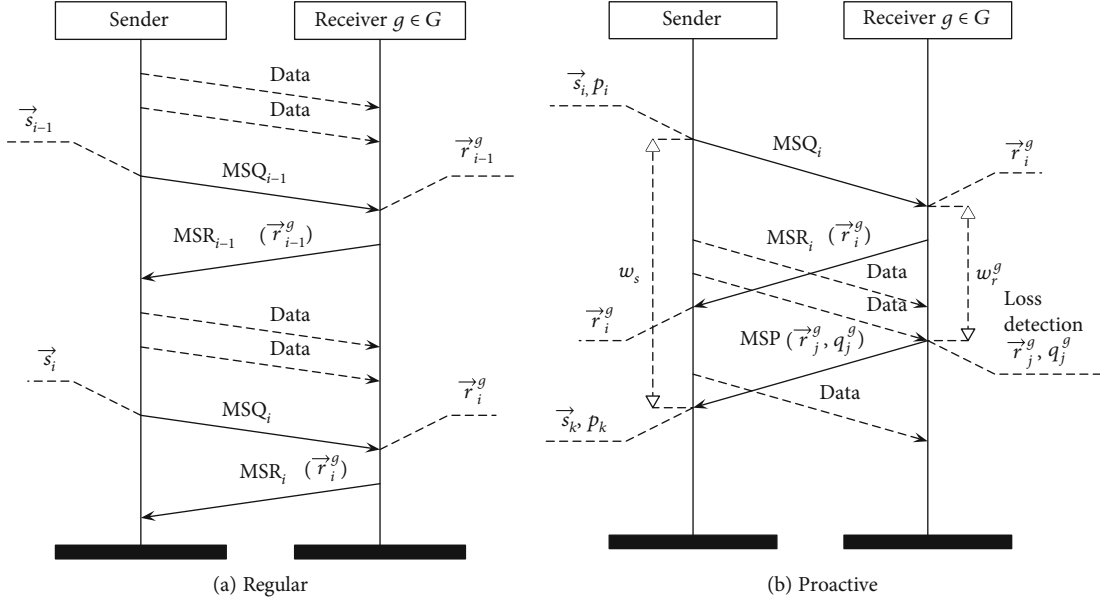


FIGURE 3: Multicast statistics query/report for aggregated feedback collection in EWRiM.

state of the frame counter vector as  $\vec{s}_i$ . The receiver responds to such a query with a *multicast statistics report (MSR)* containing its current receive counter vector  $\vec{r}_i^g$  and, if available, the average received SNR.

Based on this mechanism, the sender can calculate the number of sent and received frames per transmission rate for a specific interval as  $\Delta \vec{s}_{i-1,i} = \vec{s}_i - \vec{s}_{i-1}$  and  $\Delta \vec{r}_{i-1,i} = \vec{r}_i^g - \vec{r}_{i-1}^g$ , respectively. In case a query or response frame gets lost, the sender can simply send a new query (with a new sequence number) and then, for example, calculate the difference between the vectors at  $i-2$  and  $i$ . This also applies to multiple frames lost in a row, but for the simplicity of description, we will use the default case of  $(i-1, i)$  in our notation hereafter, without loss of generality.

Once getting the response, the sender can calculate the PDR vector  $\vec{d}_i^g$  of a group member for the previous interval as

$$\vec{d}_i^g = \Delta \vec{r}_{i-1,i}^g \oslash \Delta \vec{s}_{i-1,i}, \quad (1)$$

where  $\oslash$  stands for the *Hadamard*, or element-wise, division. Afterwards, the average PDR vector  $\vec{\delta}^g$  (using EWMA with  $\alpha_d$  as smoothing factor) is calculated as

$$\vec{\delta}_i^g = \alpha_d \vec{d}_i^g + (1 - \alpha_d) \vec{\delta}_{i-1}^g. \quad (2)$$

We use the notation  $\vec{\delta}^g$ , without the subscript, to always refer to the latest. On each update by one of the receivers to the sender, a new transmission rate selection is initiated.

For this, the minimum PDR vector  $\vec{\delta}^*$  is calculated as

$$\vec{\delta}^* : \vec{\delta}(t) = \min_{g \in G} \vec{\delta}^g(t), \quad \forall t \in T. \quad (3)$$

After this, the set of suitable transmission rates  $T^*$ , providing at least a PDR as specified by the threshold  $\tau_d$ , can be updated:

$$T^* := \left\{ t \mid t \in T \wedge \delta^*(t) \geq \tau_d \wedge \vec{s}(t) \geq s_{\min} \right\}. \quad (4)$$

For improving the rate selection stability at the beginning, a minimum number of transmitted sample frames is required, which is expressed by  $s_{\min}$ , before considering a rate selection. This mechanism acts as a filter preventing the rate selection mechanism from “overrating” a specific transmission rate based on just a few successful transmissions. Increasing the value of  $s_{\min}$  increases the stability at the expense of slower adaptivity. Finally, the transmission rate  $t^*$  for the flow is selected as the rate that provides the highest throughput out of  $T^*$ , where  $b(t)$  represents the throughput achieved at transmission rate  $t$ :

$$t^* = \arg \max_{t \in T^*} b(t). \quad (5)$$

**3.3.3. Proactive Feedback.** During the multicast rate selection process based on rate sampling and aggregated statistics, it could happen that the currently selected transmission rate is too high that it results in extensive packet losses at one of the receivers. Since the proposed approach would eventually reduce the transmission rate in such a case, but only after a certain amount of time, we introduce an additional mechanism based on proactive reporting from receivers in order to more quickly recover from such states, thus improving the adaptivity of the protocol.

In case a receiver detects a high packet loss in the FEC decoder (as described in Section 3.2), it initiates a *Multicast Statistics Proactive Report (MSP)* to the sender, without waiting for the explicit request from the sender. This mechanism is illustrated in Figure 3(b). It should be noted that the time

index  $i$  here refers to a sequence number, respectively, the point in time when the sender transmits the corresponding MSQ message, whereas  $j$  and  $k$  represent the times when the receiver detects the packet loss and initiates the proactive feedback, respectively, when this message arrives the sender. Since the packet counter vectors  $\vec{s}$  and  $\vec{r}^g$  are not synchronized in this case and a significant amount of packets may have passed the sender counter without yet being received (due to channel allocation delays and queuing), we use an estimation for the PDR as explained in the following.

For proper interpretation of the proactive reports from receivers, the sender needs to store the last transmitted data frame sequence number  $p_i$ , in addition to the packet counter vector  $\vec{s}_i$ . On detecting a significant loss at time  $j$ , the receiver sends an MSP message containing its current receiver counter vector  $\vec{r}_j^g$  and the sequence number of the last received data frame  $q_j^g$ . When this message is received at time  $k$ , the sender can calculate the current sender window size  $w_s = p_k - p_i$  (as the number of data frames sent between times  $i$  and  $k$ ) and the receiver window size  $w_r^g = q_j^g - p_i$  (as the number of data frames that should have been received until time  $j$ ). The quotient of these values is interpreted as the expected share of frames that could have been received compared to the sent frames before time  $k$ . It should be noted that  $w_s$  and  $w_r^g$  contain the number of uncoded source frames during the corresponding intervals. The number of transmitted frames is larger by the factor of the inverse FEC code rate  $1/R$ , but since that factor is the same for  $w_s$  as well as for  $w_r^g$ , it can be left out for calculating the quotient.

This estimation is based on the assumption that the distribution of the selected transmission rates did not change between times  $i$  and  $j$ , which is not necessarily the case (e.g., if the selected rate has changed in between). Therefore, only the rate  $t'$ , at which the largest number of packets had been sent, is used for the new rate selection decision later on, since it is most likely to have caused the detected high level of packet loss:

$$t' = \arg \max_{t \in T} \Delta \vec{s}_{i,k}(t). \quad (6)$$

Based on the available information, the PDR vector is then calculated as

$$\vec{d}_k^g = \Delta \vec{r}_{i,j}^g \frac{w_r^g}{w_s} \Delta \vec{s}_{i,k}. \quad (7)$$

Since a proactive report is sent only in emergency situations, the EWMA filtering is not used at this point and  $\vec{\delta}_k^g$  is calculated in this way:

$$\vec{\delta}_k^g(t) = \begin{cases} \vec{d}_k^g, & t = t', \\ \vec{\delta}_i^g, & t \neq t'. \end{cases} \quad (8)$$

In case  $w_s$  is too small for any reasonable estimation, the described process is performed based on the previous interval counters ( $\vec{s}_{i-1}$ ,  $p_{i-1}$ , and  $\vec{r}_{i-1}^g$ ). This threshold for the required sender window size  $w_{\min}$  is set by default to the half of the FEC window size. The remaining rate selection process is identical to the previously described method.

## 4. Implementation and Operation Details

The multicast rate control approach described in the previous section has been implemented in ns3 [30] version 3.29. Even though our model has been realized as a separate module, some minor modifications within the ns3 code base of the 802.11 stack were necessary. In particular, it required additional hooks in the rate control handling of multicast frames but also some extended tracing. Since the current approach covers only the modes 802.11a/b/g, the implementation is currently restricted to the same versions. The FEC layer has been implemented using the *Kodo* [28] and *Kodo Slide* [27] libraries.

For the simulation, we used the Yans 802.11 models of ns3 on its default parameters. The network nodes were configured using the ad hoc MAC in 802.11g mode. Relevant parameters for node positioning, multicast rate control, and FEC are presented in Table 1. Our implementation and further details are available online (<https://doi.org/10.14279/depositonce-8582>).

**4.1. Evaluation Metrics.** Although the actual comparative evaluation of EWRiM is presented later in Section 5, the evaluation metrics are introduced here to also facilitate the detailed protocol behavior and performance traces in this section. We use three intuitive metrics (*packet loss ratio*, *packet delay*, and *channel utilization*), which are explained in the following.

**4.1.1. Packet Loss Ratio.** From higher layer perspective, the *packet delivery ratio* (PDR) at the transport layer (UDP) is the most important criteria. For our purposes, we use the corresponding *packet loss ratio* (PLR) (with  $PLR = 1 - PDR$ ) since it allows a more intuitive visualization for small loss rates, which the rate control aims at. This PDR, resp., PLR, has to be differentiated from the PDR used within the rate control as described in Section 3.3, which is calculated on the MAC frames below the intermediate FEC layer.

For calculating the application layer PLR, the number of sent and received packets is compared for each receiver over the whole simulation time. In the comparisons that follow in Section 5, the node-specific PLR values are aggregated as an average among all receiver nodes and the individual simulation runs.

**4.1.2. Packet Delay.** A second application layer criterion is the *packet delay* between transmission and reception. This delay is measured at the transport layer interface (UDP socket) and covers the whole transmission process including the queuing at the sender, the wireless channel access, the MAC layer retransmissions in the unicast case, and the FEC decoder queuing. Thus, a high delay cannot be traced to a singular



TABLE 1: Summary of the simulation parameters used in the study.

Parameter	Section 4.2 Without SNR limiting	Section 4.2 With SNR limiting	Section 4.3 Dynamic environment	Section 5.1 Music streaming	Section 5.2 Large frames	Section 5.3 IoT
Number of receivers	20	20	2	1 to 50	1 to 50	1 to 50
Min. distance (m)	10	10	10	10	10	10
Max. distance (m)	70	70	10 to 61	10 to 100	10 to 100	10 to 100
Packet interval (ms)	10	10	2.5	20	15	1
UDP Payload size (bytes)	1024	1024	128	332	1450	16
FEC block size ( $k$ )	4	4	4	4	4	4
FEC redundancy ( $n - k$ )	1	1	1	1	1	1
FEC window size	32	32	32	32	32	32
EWriM PDR threshold ( $\tau_d$ )	0.95	0.95	0.95	0.95	0.95	0.95
EWriM RX stat. EWMA factor ( $\alpha_d$ )	0.5	0.5	0.75	0.5	0.5	0.5
EWriM FEC decoder proactive report trigger threshold	16	16	8	8	8	8
EWriM min. sender window size ( $w_{\min}$ )	16	16	16	16	16	16
EWriM min. required sample frames per rate ( $s_{\min}$ )	0	3	3	3	3	3
EWriM SNR sampling limit tolerance (dB)	$\infty$	3	3	3	3	3

source without additional data, but it can be quite important for some types of applications and can serve as an indicator for the network conditions. In our environment, we measure the delay for each individual successfully transferred packet on the transport layer. These delay values are used for investigating the behavior of a single node as in Section 4.2 and in an aggregated form similar to the PLR in the general comparison.

**4.1.3. Channel Utilization.** Since we assume a simple scenario with a constant packet rate on the application layer that does not adapt to the current network conditions (in order to keep our focus on link layer performance), we use *channel utilization* as the main metric for the performance. Therefore, we aim at reducing the channel utilization for a given traffic pattern.

Within this context, channel utilization is defined as the ratio of the channel allocation time used by a node for transmitting its frames to the total time of a given interval. The channel allocation therefore includes not only the frame itself but also the interframe spaces and the corresponding acknowledgment in case of unicast frames.

Because the current version of *ns3* does not provide a direct method for measuring the channel utilization, we had to implement this on our own. In this scenario, we measure the sum of all channel allocation of any node, which also includes frames transmitted by the receivers. For our purpose, the channel allocation includes the transmission time itself, but also the fixed time slots belonging to that (*SIFS* + *Ack-Slot* + *DIFS* for unicast, only *DIFS* for broadcast and

multicast), but without the backoff window. Since the channel utilization can only be calculated for a specified time interval, we use a slot of 100 ms for the individual investigation in Section 4.2 and the whole simulation duration with further aggregations, as for the other metrics in the later comparisons.

**4.2. Detailed Analysis in Static Environment.** In this section, we explain the behavior for *EWriM* based on a selected example. One sender node is generating a UDP flow (1024 bytes payload, 10 ms packet interval) beginning immediately at the simulation start. A group of 20 receivers is located around the sender in a spiral layout (as shown in Figure 4(a)) at distances between 10 m and 70 m. This layout provides different channel conditions at all receivers, which is a complex case for the multicast rate control. The group membership announcement interval is set to 1 s with a random offset for each receiver.

Figure 5 visualizes some essential aspects of *EWriM* in two different variations. The selected transmission rate is plotted as an orange line over the simulation time ( $x$ -axis) with the rate on the left  $y$ -axis. Additionally, the individual sampling frames are shown as small green dots. The blue line, which corresponds to the right  $y$ -axis, shows the minimum SNR reported to the sender.

At the beginning, the packets are transmitted on the default rate of 1 Mbps since the rate control has not taken any other decision (not plotted in the graph), but a few sampling frames are sent on rates up to 54 Mbps. After about 1 s, the receiver statistics collection starts (also including the

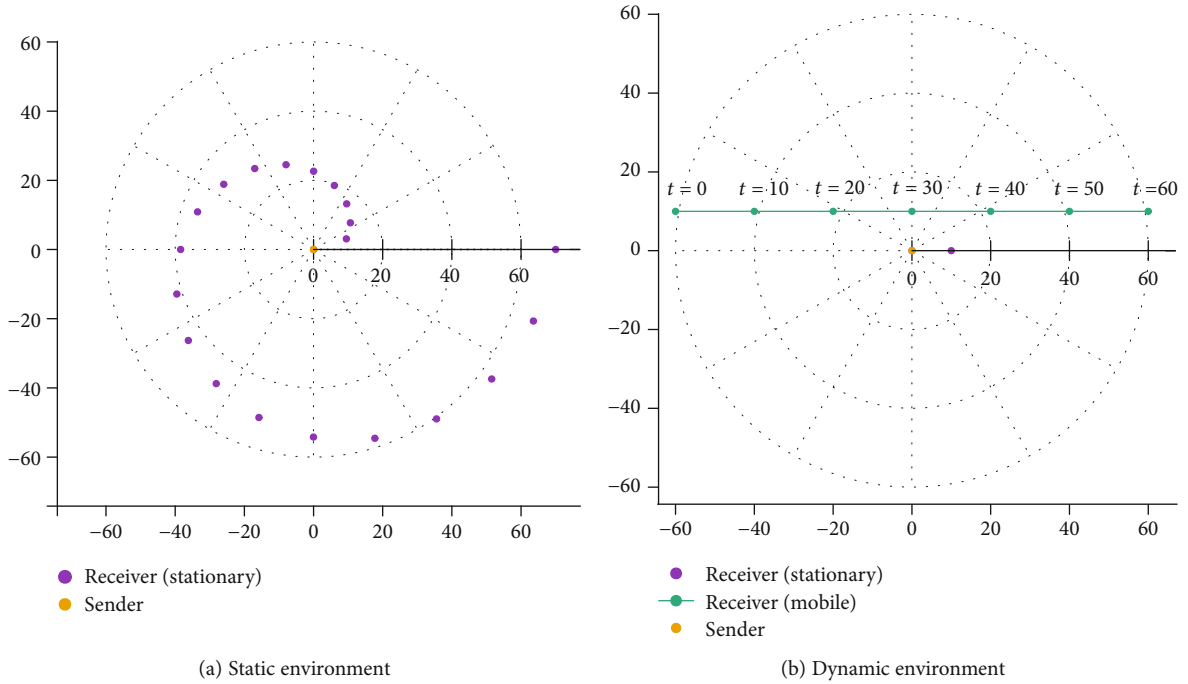


FIGURE 4: Node positions in the example networks used for presenting the detailed operation of *EWriM*.

SNR) and the rate control is beginning to take active decisions (starting with the default rate due to insufficient available feedback information). In Figure 5(a), the sampling rate selection uses the full range of available rates, while it is limited in Figure 5(b) by the minimum reported SNR with an additional tolerance of 3 dB. Additionally, the minimum number of required samples per rate is set to 0 in the first example (while it is 3 by default) for provoking some rate misselections to demonstrate some specific effects. For avoiding the rate control decision being jammed by unresponsive nodes, the rate control decision starts when feedback from 70% of the receiver group nodes is received, which is here the case at about 1.8 s. In the first case (without SNR limiting), the sampling continues on the full range of available rates independently of the reported SNR. After switching to 12 Mbps first, the rate selection mechanism further selects the rates 36 Mbps at 2.2 s and then 24 Mbps at 3.0 s. Since these rates turn out to be too high for some receivers, they cause a high packet loss, which is then detected in the FEC decoding process.

As described in Section 3.3.3, the affected receivers send proactive feedback messages in such cases, which prompt the sender for an immediate rate reduction. The effects are visualized for a selected node using the packet delay and decoder queue length (Figure 6(a)) and the channel utilization (Figure 7(a)). The packet delay (orange plot) first increases continuously up to about 120 ms, which suggests a light overload situation (Figure 6(a)). At 1.8 s, when the rate control increases the transmission rate, the delay drops immediately. As the selection process increases the rate later (at 2.2 s and 3.0 s) beyond the limits, the decoder queue length rises up to the window size. Simultaneously, the packet delay also grows (up to about 320 ms), but this time not caused by queuing on the transmission but on the FEC

decoder due to unrecoverable lost packets. The maximum decoding delay can be calculated by the FEC window size (32) multiplied by the packet interval (10 ms), which matches exactly the observed values.

The proactive feedback messages are triggered when the decoder queue length reaches 50% of the window size. Even on an instantaneous reaction, this mechanism cannot provide full recovery of lost packets if the amount of missing packets exceeds the redundancy degree of the FEC so that some frames may get lost. Figure 8(a) shows the individual packet losses with the node distance at the y-axis. Since all nodes are located at different distances (due to the spiral layout), the distance values on the y-axis also refer to the individual receiver nodes. Layer 2 (L2) losses are plotted as small green dots and layer 4 (L4) losses as larger orange dots. As can be seen in the plot, most losses correlate with distance, with the frames being received by some closer nodes while being lost on other nodes further from the sender. This can be assumed to be mostly caused by the selected transmission rate on sampling frames. In some cases, L2 frames are lost on some nodes while they have successfully been received on more distant nodes (e.g., at 1.3 s). Such a behavior is probably caused by other effects, for example, a packet collision. During the two erroneous rate selections, the L2 loss is too high for being recovered by the FEC layer, in which case L4 loss is also observed on the plot. In this example, a maximum of 14 packets out of 2000 were lost at a single receiver due to those two wrong rate selection decisions, which results in a total PLR of just 0.7%.

Finally, the belonging channel utilization, as explained in Section 4.1.3, is displayed in Figure 7(a). For this plot, the channel allocation times are aggregated in 100 ms intervals and then divided by the interval length. As can be seen in the diagram, the utilization starts with a level close to 1

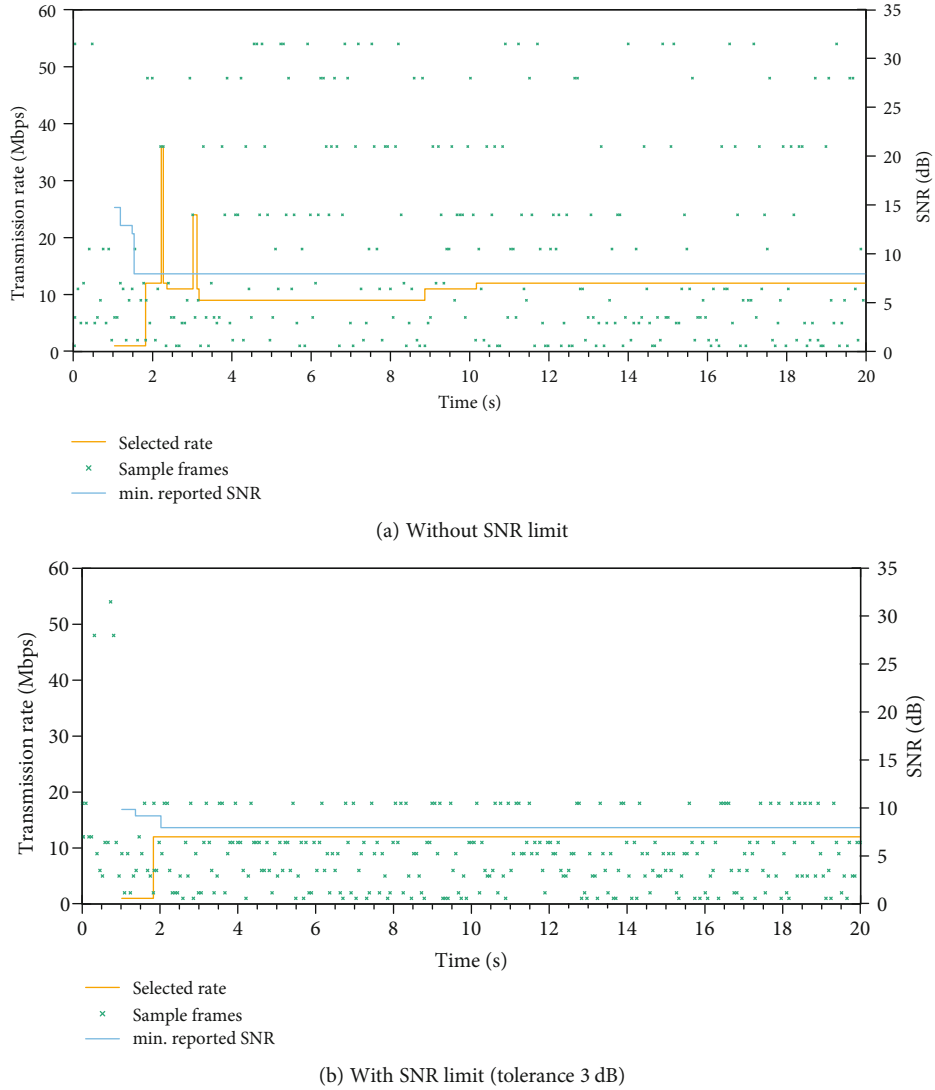


FIGURE 5: Rate selection and sampling traces of *EWRiM* for the static environment examples.

(which confirms the overload situation) and then drops to 10% to 20% when the transmission rate increases.

In the second case, the SNR sample rate limitation has been enabled and the minimum sample number per rate is set to default value of 3. As shown in Figure 5(b), after the initial phase, the rate selection switches to 12 Mbps at 1.8 s. Since the sampling concentrated on more relevant rates and provides more valid data, the rate selection remains stable for the whole duration (Figure 5(a)). After switching the transmission rate, the delay plot (Figure 6(a)) shows only some short peaks caused by L2 packet losses, which could be recovered by the FEC layer (as shown in Figure 8(a)). This can be seen in the decoder queue length (green graph), which shows the corresponding peaks. In this case, all UDP packets were received successfully and the decoder queue length reached a maximum of 8. The channel utilization (Figure 7(b)) is comparable to the first case.

**4.3. Detailed Analysis in Dynamic Environment.** The previous scenario demonstrated the general behavior of *EWRiM* in a

static environment. Since a rate control must be able to handle dynamic conditions, the adaptivity will be shown in a scenario with a mobile receiver. The layout, as shown in Figure 4(b), contains a stationary receiver at 10 m and a second one starting at about 61 m with  $2 \text{ ms}^{-1}$ . During the simulation time of 60 s, the mobile receiver passes the sender at a minimum of 10 m before it reaches again a distance of 61 m. Here, we use a smaller payload of 128 bytes at a short packet interval of 2.5 ms. The remaining parameters can be found in Table 1.

As in the previous case, the selected rate, the sampling frames, and the reported SNR are plotted in Figure 9. After the initial phase, a rate of 18 Mbps is selected while the sampling also covers 24 Mbps. As the mobile receiver gets closer, the minimum reported SNR increases and the sender starts sampling with higher rates (at 8.2 s, 16.0 s, and 17.0 s). After a sufficient PDR has been reported, the rate control increases the selected rate (at 9 s, 14 s, 20 s, and 21 s). Since the approaching phase results in improving conditions, the rate selection is not critical in terms of packet losses.

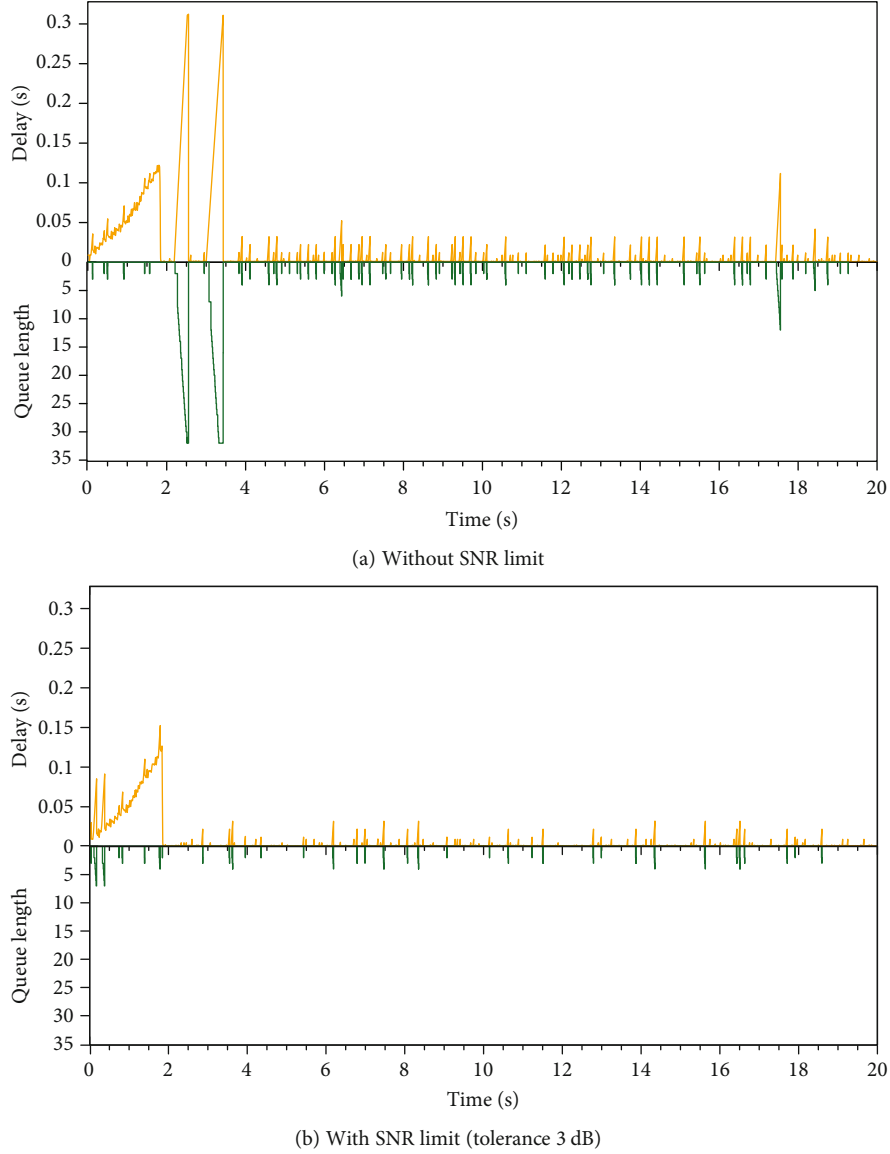


FIGURE 6: Delay and decoder queue length traces of *EWRiM* for the static environment examples.

After passing the closest point at 30s, the SNR starts decreasing. Because the regular feedback collection reacts too slow for a rate reduction before causing a significant packet loss, the proactive feedback mechanism becomes essential for the phase of declining conditions. As the plots of delay and decoder queue length in Figure 10 point out, the FEC decoder detects relevant packet losses and triggers the proactive feedback, which then leads to a rate reduction by the receiver (at 40.9 s, 42.1 s, 48.0 s, and 52.6 s). Even after the selected rate has been reduced, the sampling continues at higher rates until the reported SNR falls below particular thresholds (at 44.0 s, 46.0 s, 52.6 s, and 58.7 s).

Finally, the packet losses are plotted in Figure 11. In this example, the stationary node receives all L2 frames (with one exception at 24.5 s), so that almost all losses correspond to the mobile node.

The visible gaps in the L2 loss during the approaching phase are caused by improving conditions (which results in

less packet losses), until a higher rate is selected. As the distance increases in the second half, the L2 losses stay at a higher level since the rate control reacts on losses before reducing the rate. In this example, all L2 losses could be recovered by the FEC layer so that no L4 loss occurs.

## 5. Performance Evaluation

While the previous section focused on explaining and showcasing the detailed inner workings of *EWRiM* on example networks and nodes, in this section, we evaluate its performance and scalability under the different conditions of three realistic scenarios, in comparison with unicast and legacy multicast. A direct performance comparison to other approaches was not feasible, due to the unavailability of comparable models with public implementation details. Only few of the solutions in the literature, as covered in Section 2, address both aspects of reliability and transmission rate



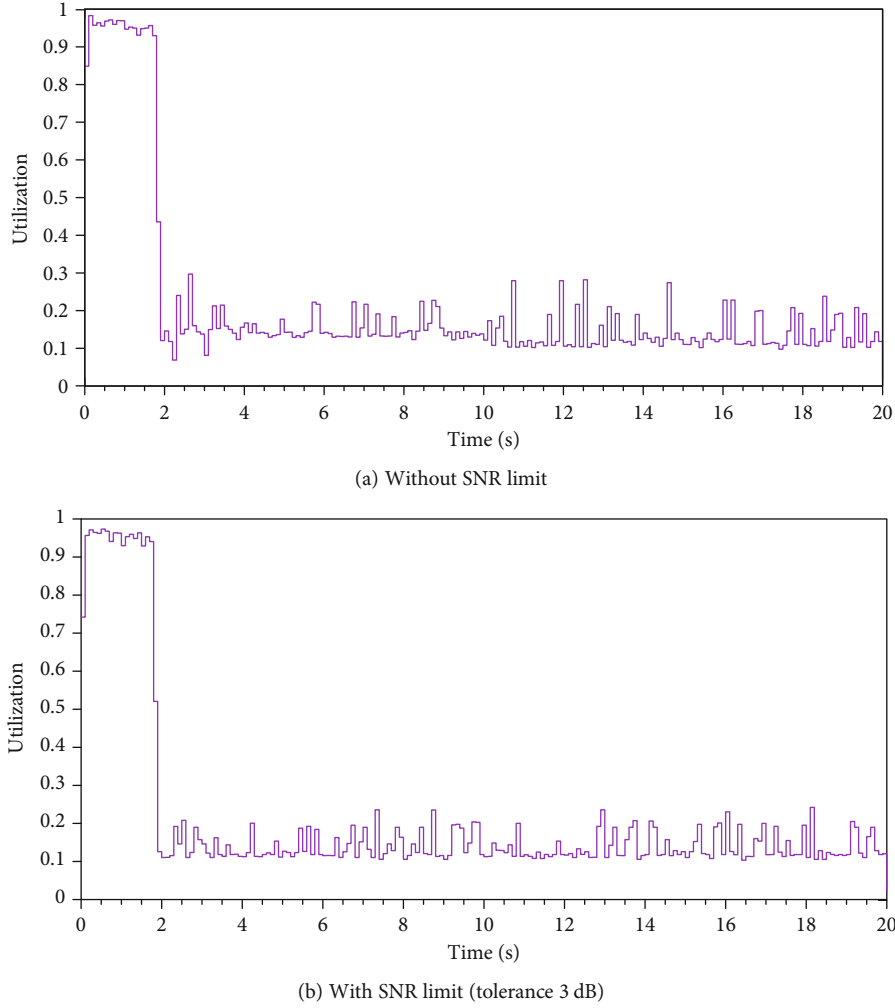


FIGURE 7: Channel utilization traces of *EWRiM* for the static environment examples.

adaption for general IP multicast traffic; many are limited to special applications like real-time video content distribution. Therefore, we confine our comparisons to the unicast and legacy multicast approaches, which are also the current state of practice for realizing reliable wireless multicast scenarios.

We employ a spiral layout for the network topology similar to Figure 4(a), but with the number of receivers (1 to 50) and the maximum distance (10 m to 100 m) as the variable parameters. Each resulting combination is simulated 5 times with *EWRiM*, legacy multicast, and unicast. In the latest case, the traffic is generated as an individual application layer flow for each receiver, but the general behavior should also be comparable to a multicast-to-unicast conversion at the MAC layer as provided by the DMS. The wireless network configuration is identical to the previous setup used in Section 4.2.

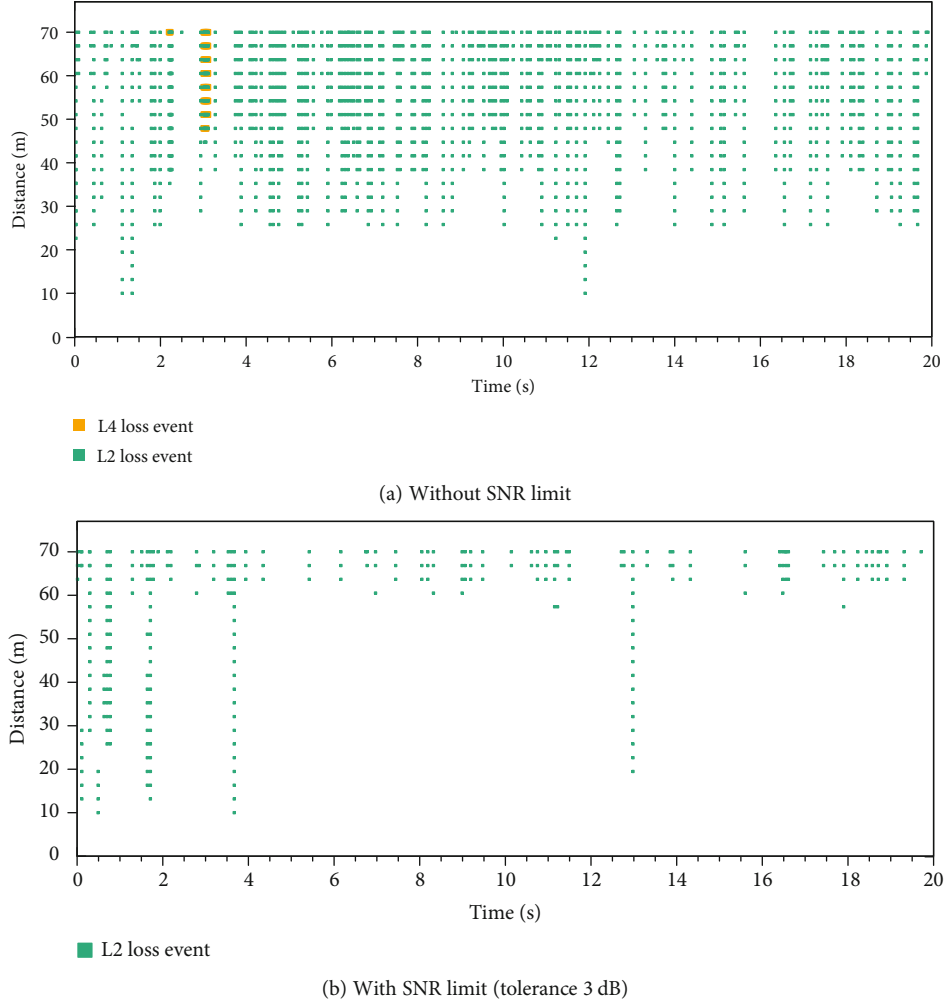
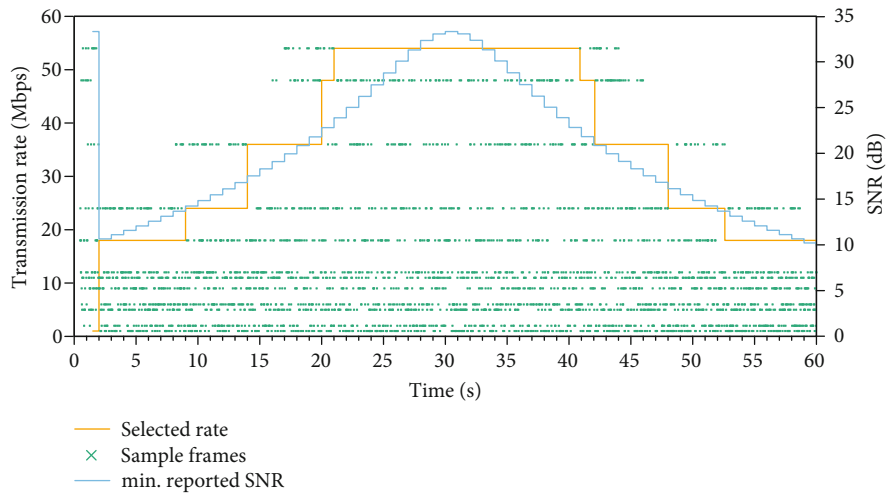
For investigating the behavior of *EWRiM* under different conditions, we adopt three application scenarios, which are described next. The corresponding parameters for *EWRiM*, FEC, and traffic patterns are listed in Table 1. Since TCP is designed for unicast communication only, we are using UDP as transport protocol for our evaluation. This also allows us to focus on the MAC layer performance; hence, we do not

consider higher layer aspects such as packet/content fragmentation, transport layer rate adaptation, and end-to-end reliability. Such additional functionalities could be provided by a multicast transport protocol like *NORM* [26].

**5.1. Music Streaming Scenario.** One of the typical scenarios for multicast applications is real-time distribution of multimedia content. A special case, for example, is the audio distribution to a multiroom speaker system. For this comparison case, we consider an audio stream with 128 kbps at a frame rate of 50 fps, which results in a packet interval of 20 ms with a UDP payload size of 332 bytes (including a RTP header of 12 bytes).

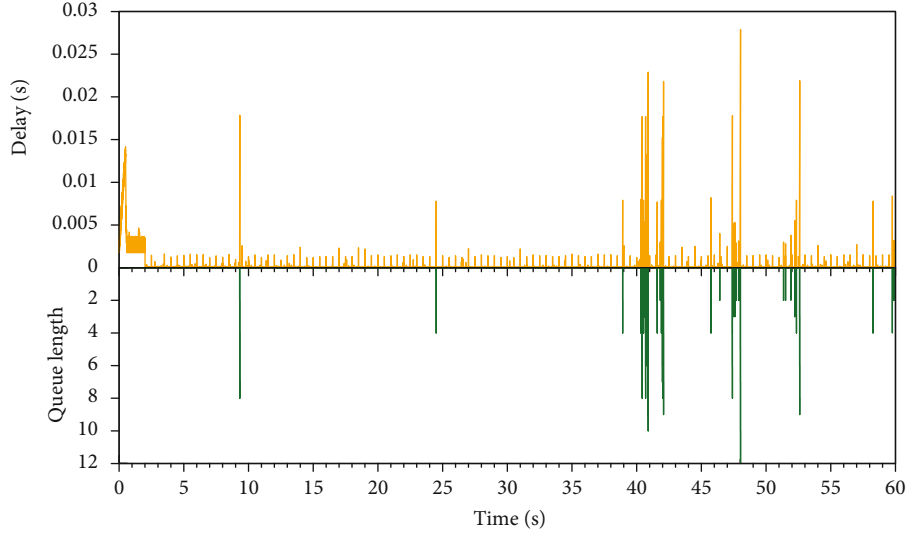
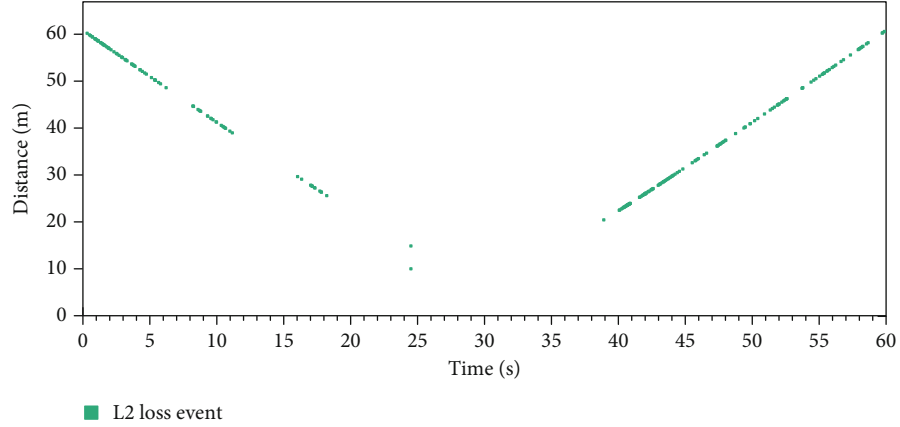
Figure 12(a) shows the packet loss ratio for this scenario. Up to 20 nodes, the PLR is close to 0 for all traffic modes, but both multicast cases show a small loss in some situations. In case of unicast traffic, the PLR rises significantly for more than 20 to 40 nodes, depending on the maximum distance, and reaches a maximum average loss of about 35%. In contrast, multicast methods are not much affected by the number of nodes and the distance, both staying close to zero.

The packet delay in Figure 12(b) shows similar characteristics. Due to its simple mechanism, the default multicast

FIGURE 8: Packet loss traces of *EWRiM* for the static environment examples.FIGURE 9: Rate selection and sampling traces of *EWRiM* for the dynamic environment example.

mode provides a constant delay close to zero at the chosen scaling, independent of the number of nodes and distance. In the *EWRiM* multicast mode, the delay shows a similar behavior with a few small exceptions. For small numbers of

nodes, the unicast transmission mode provides also a small delay, but it rises up to 0.7 s as the number of nodes increases. This increased delay correlates directly with the PLR in Figure 12(a).

FIGURE 10: Delay and decoder queue length traces of *EWRiM* for the dynamic environment example.FIGURE 11: Packet loss traces of *EWRiM* for the dynamic environment example.

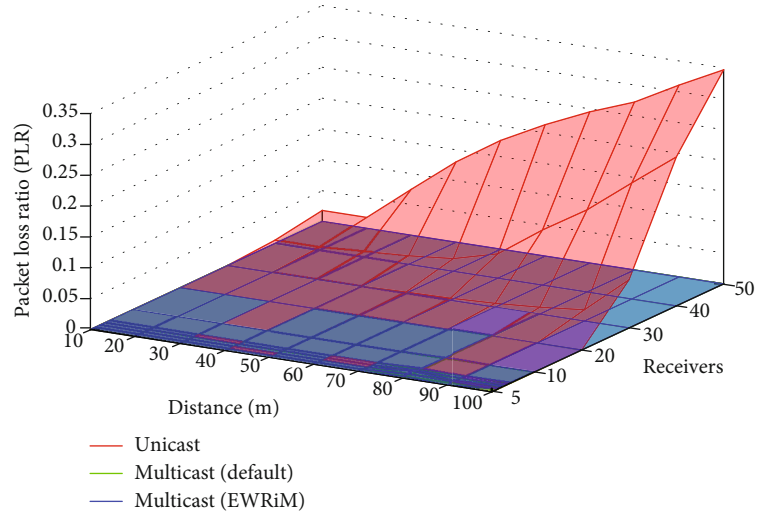
The channel utilization in Figure 12(c) highlights the differences among all three methods more significantly. As for the delay, the default multicast method shows a constant behavior, independent of receiver group size and distances, at 22%. In the unicast case, the utilization grows almost linearly with the number of receivers until it reaches a maximum of about 75%, and it also rises with the growing distance. In contrast, our approach provides significantly lower utilization than that of the default multicast mode (between 2% and 14%) and is almost independent of the number of receiver nodes.

In this scenario, both multicast modes provide scalable content distribution while the unicast case exceeds the channel capacity in the case of larger receiver group sizes. Compared to the default multicast behavior, our approach allows a significant reduction of channel utilization close to a single unicast stream.

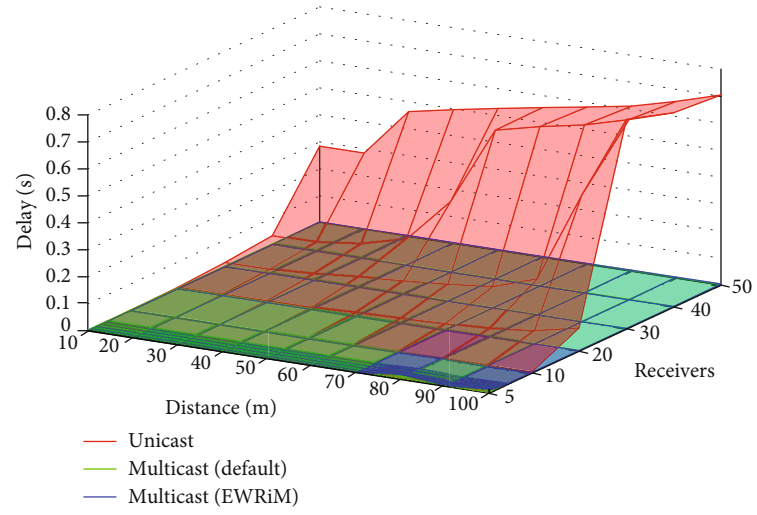
**5.2. Large Data Frame Scenario.** Another use case is the distribution of bulk data to many receivers. This is less common on multicast since it is a special application scenario that

requires an appropriate transport protocol assuring reliability and dealing with additional challenges (e.g., congestion control). A candidate for this would be the NORM protocol [26], which internally combines leader-based concepts, NAKs, FEC, and other mechanisms for achieving a reliable and scalable data transfer via multicast IP. From the perspective of our proposed protocol *EWRiM*, we can assume that such a transport protocol would try to use frames as large as allowed by the path MTU and it would adapt to a rate close to the maximum network capacity. For this evaluation, we are focusing on the rate control behavior itself, and therefore, we leave the adaptive transport protocol out and use a static payload size of 1450 bytes and a frame interval of 15 ms, which are together close to the capacity limits of the lowest transmission rate of 1 Mbps.

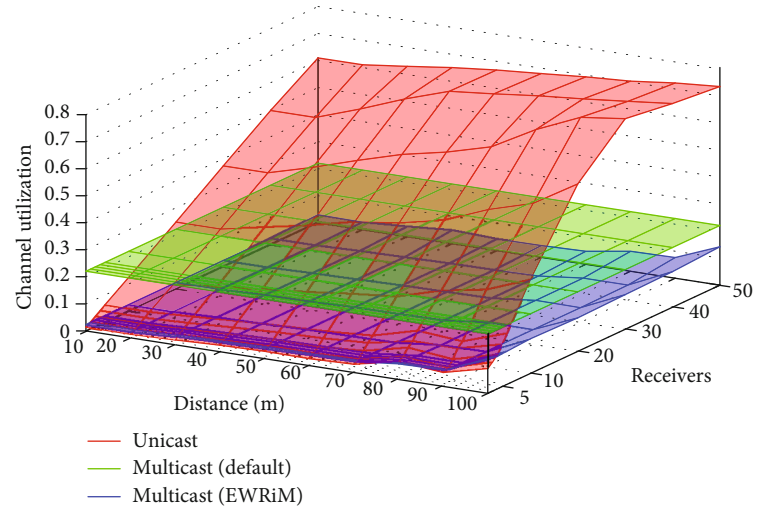
Since *EWRiM* relies on sample frames, larger frames at lower rates lead to less samples in total, as the overall number of frames decreases. This could have negative effects on the rate control behavior, which is why this scenario was chosen as an additional test case. The parameters, with the exception of frame size and rate, are similar to the previous scenario.



(a) Packet loss



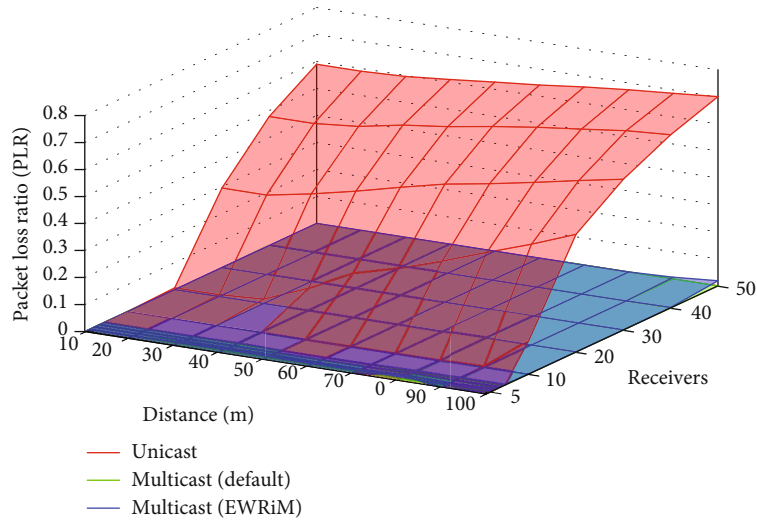
(b) Packet delay



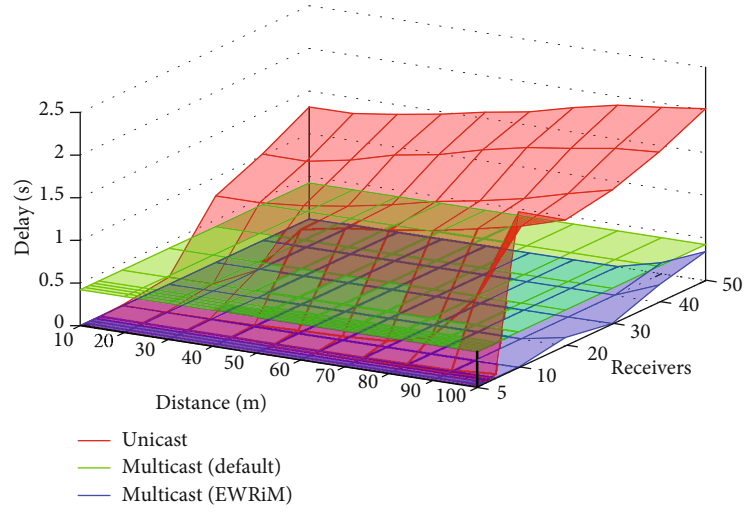
(c) Channel utilization

FIGURE 12: Comparison of EWRiM with legacy multicast- and unicast-based communication in the music streaming scenario.

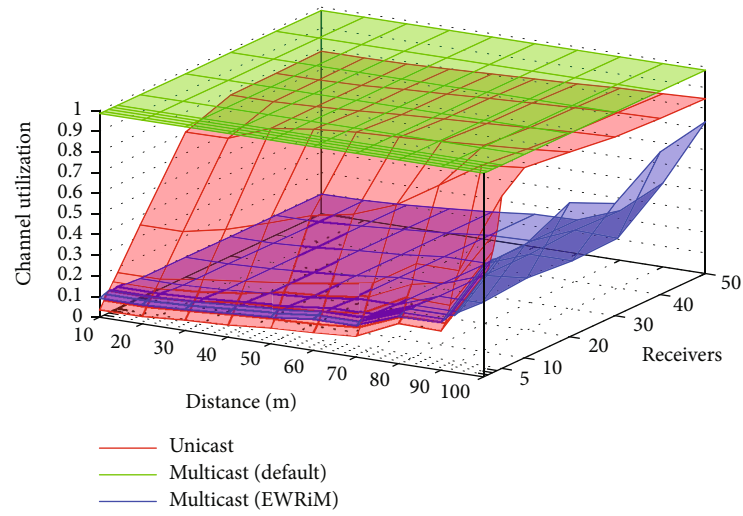




(a) Packet loss

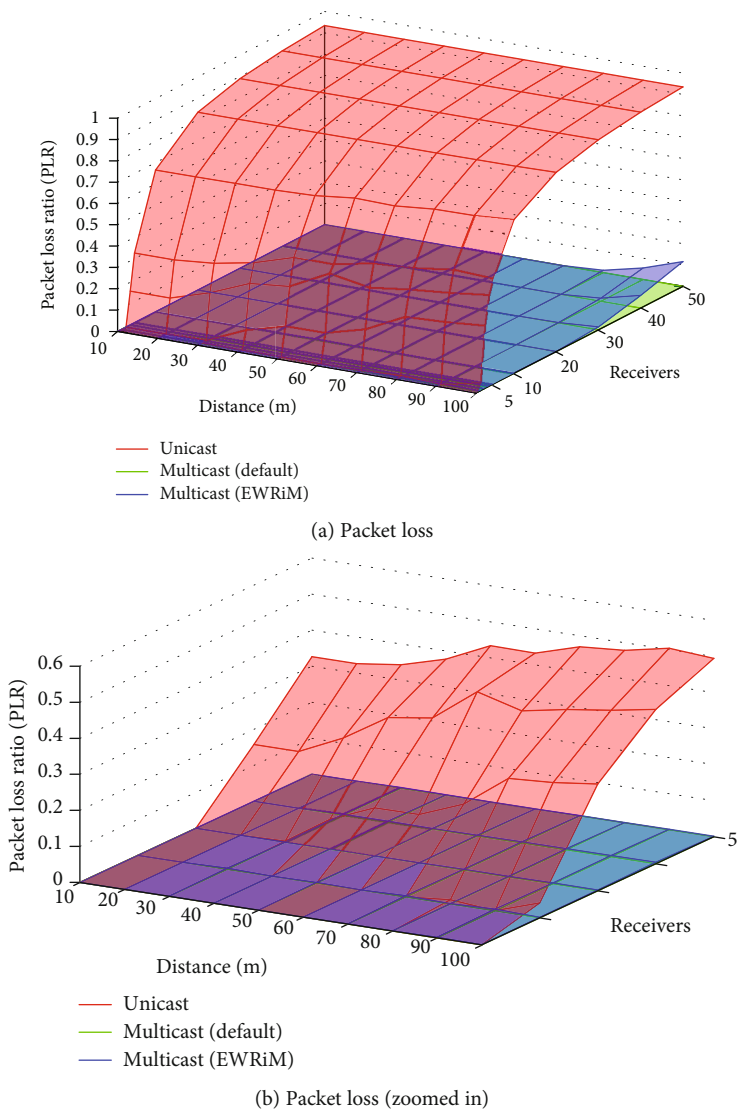


(b) Packet delay



(c) Channel utilization

FIGURE 13: Comparison of *EWRiM* with legacy multicast- and unicast-based communication in the large frame scenario.



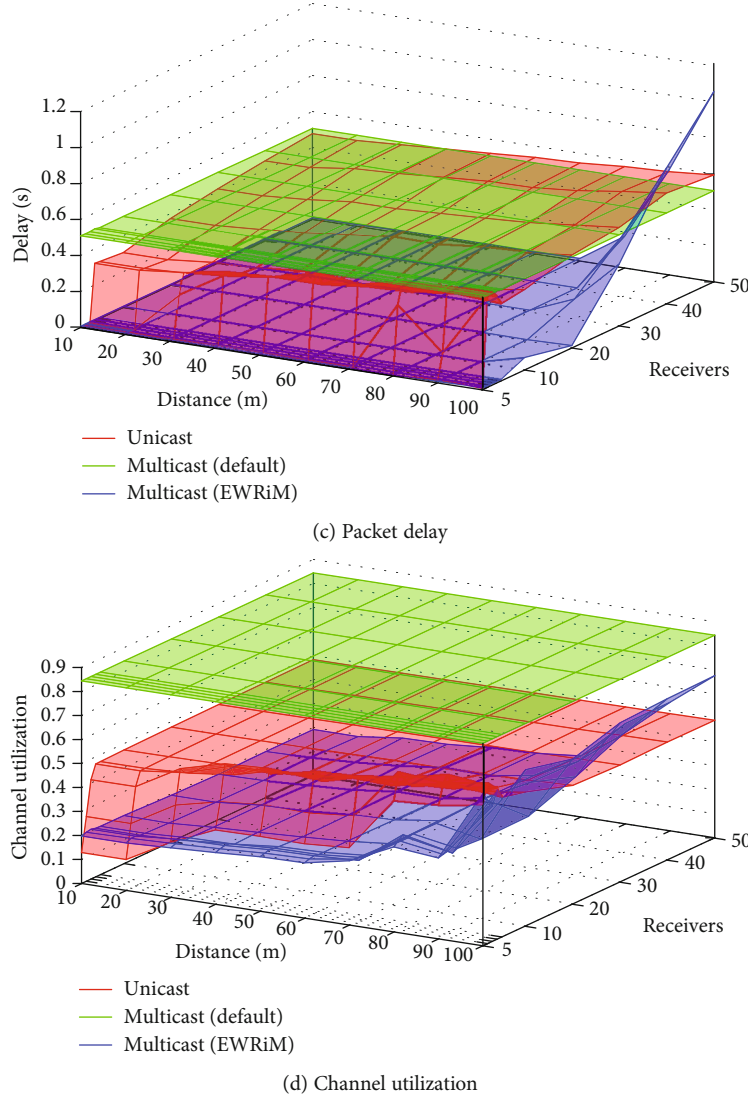


FIGURE 14: Comparison of *EWRiM* with legacy multicast- and unicast-based communication in the IoT scenario.

In this scenario, the unicast transfer starts exhibiting a significant packet loss ratio after 5 nodes for the maximum distance case and after 20 nodes for the minimum distance case, which points out the limits of the unicast transfer under such conditions (Figure 13(a)). In contrast, the packet loss on both multicast modes is not affected by number of nodes or distance. The packet delay in Figure 13(b) exhibits a corresponding behavior for the unicast case with significant delays. Since the application layer data rate has been selected to be close to the limit of the lowest transmission rate, the delay plot presents significantly large but constant delay for the default multicast mode. This pattern differs to the previous scenario (Figure 12(b)), where the data rate was far below the channel capacity at the lowest rate. When using multicast transfer with *EWRiM*, the delay is almost constant for the largest part, but shows some growth at the maximum distance with larger number of nodes.

This behavior can be explained by the channel utilization, depicted in Figure 13(c). Since the transmission rate has to be

reduced to 2 Mbps for reaching the most distant nodes and *EWRiM* generates additional overheads due to FEC and feedback data collection, it causes an increased channel utilization (up to 70%), which explains the higher delays. But even this utilization is below those of the default multicast mode and the unicast transfer (which fails to serve all receivers for larger receiver group sizes). Generally, the channel utilization in the *EWRiM* mode depends mainly on the maximum node distance (due to the usable transmission rates) and only slightly on the number of receivers (due to the aggregated feedback collection).

**5.3. IoT Scenario.** In this scenario, we consider highly frequent but small data frames, as it is a typical case in many industrial or residential IoT applications [31, 32]. The UDP payload size is set to 16 bytes at a packet interval of 1 ms, which is again close to the channel capacity of the legacy multicast mode.

The PLR in Figure 14(a) exhibits a similar general characteristic to the previous scenarios; however, since many small

packets cause higher overhead, the loss rate already starts growing significantly with increasing distance in the unicast case for 2 to 4 receivers. Due to this effect, Figure 14(b) provides a zoomed-in version of the delay plot for better visibility. In both multicast cases, the delay stays close to zero with the exception of large distances and receiver group sizes in the *EWriM* case.

The overload situation diagnosis in the unicast case is also supported by the delay plot in Figure 14(c), which shows a significant growth in the same regions. Similar to the large frame scenario, the default multicast mode causes a high delay of about 0.5 s, suggesting that data volume is close to the limit at this mode. The *EWriM* mode, in comparison, provides a significantly lower delay, close to zero, for most cases, but shows considerable growth on large receiver groups and distances.

The utilization in Figure 14(d) confirms the limit in the default multicast mode with a utilization of 84%. Compared to that, the unicast mode reaches a maximum of only 67% where it is in an overload situation and causes the largest frame loss. The reason for this difference is our utilization metric, which does not consider the backoff times on the channel allocation. Since the unicast mode transmits more frames, probably including many retransmission attempts, this plays a significant role in that case. Our *EWriM* approach achieves significantly lower channel utilization at almost the whole range except for the case of a single close receiver. On large distances, the utilization reaches a level comparable to or higher than the unicast mode (up to 86%), but still delivers the most data frames.

In contrast to the previous scenarios, *EWriM* runs into an overload situation in the case of large receiver groups at long distances, whereas the default multicast mode is still able to handle those cases. This is caused by the high number of frames, which makes it difficult to allocate slots for the feedback collection so that the rate control concept cannot work optimally. As a consequence, *EWriM* switches to the lowest transmission rate (which is also used by the default mode), but causes additional overhead by the FEC and the feedback aggregation.

## 6. Summary and Conclusion

In this article, we presented a new concept for multicast transmission rate control in IEEE 802.11 wireless local area networks. The proposed protocol embodies a novel combination of transmission rate sampling, aggregated receiver feedback, and flexible forward error correction mechanisms for improving both the performance and the reliability of multicast communications at the link layer. Based on three distinct scenarios with realistic traffic patterns, simulation results have demonstrated significant improvements over legacy multicast and unicast communications in terms of scalability and channel utilization. Since the rate control causes some management overhead for each multicast group, it is intended for handling high-throughput application flows, whereas low-rate traffic, such as service discovery or network management, can be

transmitted in the traditional way at the lowest rate. Nevertheless, the forward error correction layer could also improve reliability on such low-rate applications—possibly with some extensions like a timeout mechanism for sending redundancy frames when a source block is not completed. Compared to the *Groupcast with Unsolicited Retries (GCR-UR)* mode in the aa-amendment of the 802.11 standard, this has the potential of providing similar or better reliability at lower overhead.

In its current version, our approach covers only the legacy versions 802.11b/g/a; the extension to newer versions (802.11n and beyond) is therefore one of the essential future research directions. This is more challenging from a technical perspective, due to the larger parameter space for rate selection (modulation and coding scheme, channel width, number of spatial streams, and guard interval), but the general concepts presented in this work would still be applicable as a baseline. Another direction for further research and improvement is the automated choice of suitable parameters for the rate control algorithm and the forward error correction under different conditions. This could include adaptive mechanisms that dynamically adjust, for example, the FEC parameter code rate and window size according to the current traffic pattern and channel state conditions.

## Abbreviations

$G$ :	Set of group members
$T$ :	Set of transmission rates
$\vec{s}$ :	Absolute sender frame counter vector
$\vec{r}^g$ :	Absolute receiver frame counter vector for node $g \in G$
$\Delta \vec{s}_{i,j}$ :	Relative sender frame counter vector between requests $i$ and $j$
$\Delta \vec{r}_{i,j}^g$ :	receiver frame counter vector between requests $i$ and $j$ for node $g \in G$
$s_{\min}$ :	Minimum number of sample frames per rate required being transmitted for accepting it as a candidate
$\vec{d}_i^g$ :	PDR (packet delivery ratio) of interval $i$ for node $g \in G$
$\vec{\delta}_i^g$ :	Moving average PDR based on request $i$ for node $g \in G$ ( $\vec{\delta}^g$ always refers to the latest one)
$\alpha_d$ :	Smoothing factor (EWMA) for PDR
$\vec{\delta}^*$ :	min. PDR vector among all group members
$\vec{\delta}^*(t)$ :	min. PDR among all group members for transmission rate $t \in T$
$\tau_d$ :	Threshold for required PDR
$T^*$ :	Possible transmission rates ( $\text{PDR} \geq \text{threshold}$ )
$t^*$ :	Selected transmission rate
$b(t)$ :	Bitrate of transmission rate $t$
$P_i$ :	Sender frame counter at time $i$
$q_i^g$ :	Receiver frame counter of node $g \in G$ at time $i$
$w_s$ :	Sender window size on proactive feedback
$w_r^g$ :	Receiver window size of node $g \in G$ on proactive feedback
$w_{\min}$ :	Minimum required receiver window size for proactive feedback.



## Data Availability

The source code of our protocol implementation is available online at DepositOnce public repository of Technische Universität Berlin: doi:10.14279/depositonce-8582.

## Conflicts of Interest

The authors declare that they have no conflicts of interest.

## Acknowledgments

We acknowledge the support by the German Research Foundation and the Open Access Publication Fund of Technische Universität Berlin.

## References

- [1] A. Rahman and E. Dijk, "Group communication for the constrained application protocol (CoAP)," RFC 7390 (Experimental), 2014, <http://www.ietf.org/rfc/rfc7390.txt>.
- [2] "MQTT for sensor networks (MQTT-SN) protocol specification version 1.2," 2018, [https://www.oasis-open.org/committees/download.php/66091/MQTT-SN\\_spec\\_v1.2.pdf](https://www.oasis-open.org/committees/download.php/66091/MQTT-SN_spec_v1.2.pdf).
- [3] J. Kuri and S. K. Kasera, "Reliable multicast in multi-access wireless LANs," in *IEEE INFOCOM '99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No.99CH36320)*, pp. 760–767, New York, NY, USA, 1999.
- [4] S. K. S. Gupta, V. Shankar, and S. Lalwani, "Reliable multicast MAC protocol for wireless LANs," in *IEEE International Conference on Communications, 2003. ICC '03*, pp. 93–97, Anchorage, AK, USA, May 2003.
- [5] A. Chen, D. Lee, G. Chandrasekaran, and P. Sinha, "HIMAC: high throughput MAC layer multicasting in wireless networks," in *2006 IEEE International Conference on Mobile Ad Hoc and Sensor Systems*, pp. 41–50, Vancouver, BC, Canada, October 2006.
- [6] B.-S. Kim, S. W. Kim, and R. L. Ekl, "OFDMA-based reliable multicasting MAC protocol for WLANs," *IEEE Transactions on Vehicular Technology*, vol. 57, no. 5, pp. 3136–3145, 2008.
- [7] Y. D. Park, S. Jeon, J. Jeong, and Y. Suh, "FlexVi: PHY aided flexible multicast for video streaming over IEEE 802.11 WLANs," *IEEE Transactions on Mobile Computing*, vol. 3, p. 1, 2019.
- [8] M.-T. Sun, H. Lifei, A. Arora, and T.-H. Lai, "Reliable MAC layer multicast in IEEE 802.11 wireless networks," in *Proceedings International Conference on Parallel Processing*, pp. 527–536, Vancouver, BC, Canada, August 2002.
- [9] W. Si and C. Li, "RMAC: a reliable multicast MAC protocol for wireless ad hoc networks," in *International Conference on Parallel Processing, 2004. ICPP 2004*, pp. 494–501, Montreal, QC, Canada, August 2004.
- [10] "IEEE standard for information technology–local and metropolitan area networks–specific requirements–part 11: wireless LAN medium access control (MAC) and physical layer (PHY) specifications - amendment 8: medium access control (MAC) quality of service enhancements," in *IEEE Std 802.11e-2005 Amendment to IEEE Std 802.11, 1999 Edition (Reaff 2003)*, pp. 1–212, IEEE, 2005.
- [11] "ISO/IEC/IEEE international standard for information technology–telecommunications and information exchange between systems local and metropolitan area networks–specific requirements part 11: wireless LAN medium access control (MAC) and physical layer (PHY) specifications amendment 2: MAC enhancements for robust audio video streaming (adoption of IEEE Std 802.11aa-2012)," in *ISO/IEC/IEEE 8802-11:2012/Amd.2:2014(E)*, pp. 1–168, IEEE, 2014.
- [12] C. Chao, C. Chou, and H. Wei, "Pseudo random network coding design for IEEE 802.16m enhanced multicast and broadcast service," in *2010 IEEE 71st Vehicular Technology Conference*, pp. 1–5, Taipei, Taiwan, May 2010.
- [13] S. Wunderlich, F. Gabriel, S. Pandi, F. H. P. Fitzek, and M. Reisslein, "Caterpillar RLNC (CRLNC): a practical finite sliding window RLNC approach," *IEEE Access*, vol. 5, pp. 20183–20197, 2017.
- [14] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris, "Link-level measurements from an 802.11B mesh network," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4, pp. 121–132, 2004.
- [15] S. W. Kim and B.-S. Kim, "Rate-adaptive MAC protocol for wireless multicast over OFDMA-based MANETs," *Wireless Personal Communications*, vol. 56, no. 4, pp. 675–692, 2011.
- [16] J. Villalón, Y. Seok, T. Turletti, P. Cuenca, and L. Orozco-Barbosa, "ARSM: auto rate selection multicast mechanism for multi-rate wireless LANs," in *Personal Wireless Communications. PWC 2006*, pp. 239–250, Springer, Berlin, Heidelberg, 2006.
- [17] IEEE Computer Society, "IEEE standard for information technology–telecommunications and information exchange between systems local and metropolitan area networks–specific requirements part 11: wireless LAN medium access control (MAC) and physical layer (PHY) specifications," in *IEEE Std 802.11-2012 (Revision of IEEE Std 802.11-2007)*, pp. 1–2793, IEEE, 2012.
- [18] "IEEE standard for information technology–telecommunications and information exchange between systems local and metropolitan area networks–specific requirements - part 11: wireless LAN medium access control (MAC) and physical layer (PHY) specifications," in *IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012)*, pp. 1–3534, IEEE, 2016.
- [19] S. Paris, N. Facchi, and F. Gringoli, "Chapter 3 - rate adaptation algorithms for reliable multicast transmissions in wireless LANs," in *Modeling and Simulation of Computer Networks and Systems*, M. S. Obaidat, P. Nicopolitidis, and F. Zarai, Eds., pp. 55–94, Morgan Kaufmann, Boston, MA, USA, 2015.
- [20] E. Coronado, R. Riggio, J. Villalón, and A. Garrido, "Joint mobility management and multicast rate adaptation in software-defined enterprise WLANs," *IEEE Transactions on Network and Service Management*, vol. 15, no. 2, pp. 625–637, 2018.
- [21] E. Coronado, R. Riggio, J. Villalón, and A. Garrido, "Efficient real-time content distribution for multiple multicast groups in SDN-based WLANs," *IEEE Transactions on Network and Service Management*, vol. 15, no. 1, pp. 430–443, 2018.
- [22] V. Gupta, C. Gutterman, Y. Bejerano, and G. Zussman, "Experimental evaluation of large scale WiFi multicast rate control," in *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, pp. 1–9, San Francisco, CA, USA, 2016.
- [23] Y. Shin, G. Lee, J. Choi, J. Koo, S. Lee, and S. Choi, "InFRA: interference-aware PHY/FEC rate adaptation for video

- multicast over WLAN,” in *2017 14th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, pp. 1–9, June 2017.
- [24] Linux Wireless Kernel Community, “Linux mac80211 minstrel rate control,” 2017, <https://wireless.wiki.kernel.org/en/developers/documentation/mac80211/ratecontrol/minstrel>.
  - [25] T. Geithner and F. Sivrikaya, “Adaptive reliable multicast in 802.11 networks,” in *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, Barcelona, Spain, April 2018.
  - [26] B. Adamson, C. Bormann, M. Handley, and J. Macker, “NACK-oriented reliable multicast (NORM) transport protocol,” RFC 5740 (Proposed Standard), 2009, <http://www.ietf.org/rfc/rfc5740.txt>.
  - [27] Steinwurf ApS, “Kodo-slide, a sliding window ECC algorithm,” 2018, <https://kodo-slide.steinwurf.com/>.
  - [28] M. V. Pedersen, J. Heide, and F. H. P. Fitzek, “Kodo: an open and research oriented network coding library,” in *NETWORKING 2011 Workshops*, V. Casares-Giner, P. Manzoni, and A. Pont, Eds., pp. 145–152, Springer, Berlin, Heidelberg, 2011.
  - [29] S. Wunderlich, J. A. Cabrera, F. H. P. Fitzek, and M. Reisslein, “Network coding in heterogeneous multicore IoT nodes with DAG scheduling of parallel matrix block operations,” *IEEE Internet of Things Journal*, vol. 4, no. 4, pp. 917–933, 2017.
  - [30] The NS-3 Project, “Network simulator 3,” 2018, <http://www.nsnam.org/>.
  - [31] C. Akpolat, D. Sahinel, F. Sivrikaya, G. Lehmann, and S. Albayrak, “Chariot: an iot middleware for the integration of heterogeneous entities in a smart urban factory,” in *Position Papers of the 2017 Federated Conference on Computer Science and Information Systems*, pp. 135–142, Prague, Czech Republic, 2017.
  - [32] D. Sahinel, C. Akpolat, M. A. Khan, F. Sivrikaya, and S. Albayrak, “Beyond 5g vision for iolite community,” *IEEE Communications Magazine*, vol. 55, no. 1, pp. 41–47, 2017.